Customizing Xcode for Your Development Workflow

Session 418

These are confidential sessions—please refrain from streaming, blogging, or taking pictures.

Customizing Xcode for Your Development Workflow

Andrew Pontious
Xcode IDE Manager

Goals

- Tune Xcode to your liking
- Show where and how
- Make you more efficient

Agenda

- Text editing
- Building and debugging
- Source control management and snapshots

Agenda

- Text editing
- Building and debugging
- Source control management and snapshots

Text Editing

Dave Ewing
Xcode Engineer

Text Editing

- Editing efficiently
- Customizing the editor

Editing Efficiently

- Code Sense
- Text macros
- Key binding preferences
- Navigation
- Quick Help

Code Sense

- Code completion improvements in Snow Leopard
  - Subtle changes to the UI
  - More aware of context
  - Performance
- Auto-insertion of opening square brackets

Code Completion

- Three “modes” of operation
  - In-line
  - Next Completion (“cycling”)
  - Pop-up list

Code Completion

- Three “modes” of operation
  - In-line
  - Next Completion (“cycling”)
  - Pop-up list
  - In-line pop-up

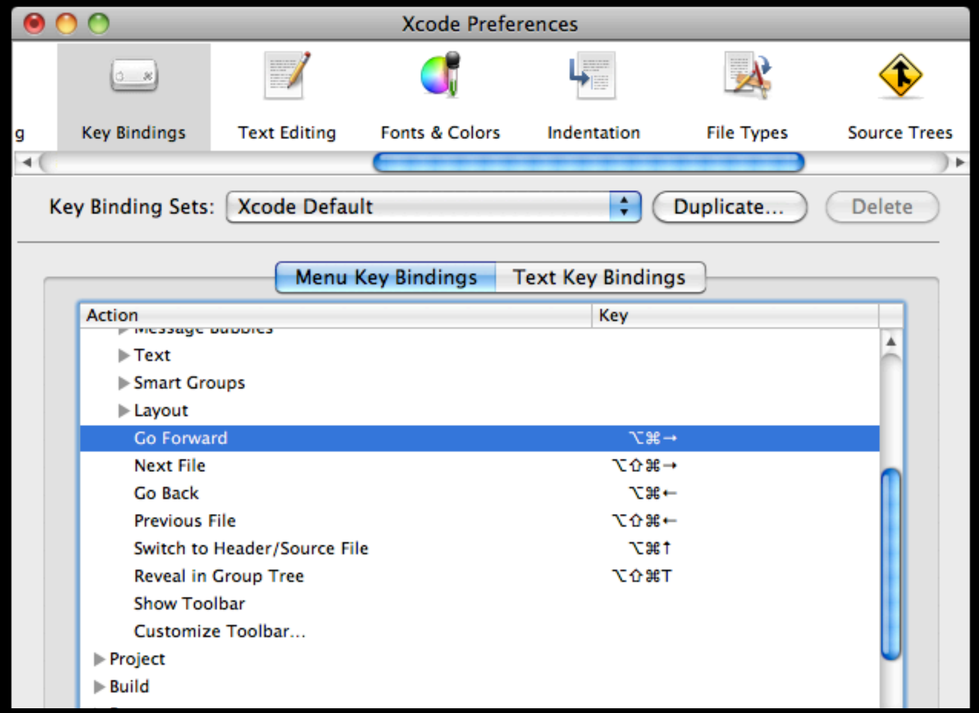
```
$ defaults write com.apple.Xcode \
  XCCodeSenseAutoSuggestionStyle List
```

Text Macros

- Little snippets of code
- Can “wrap” the current selection
- Inserted via code completion
- Available in the Edit menu
  - Assign key bindings for your favorites

Key Bindings

- Menu items
  - Text macros
- Text editing commands



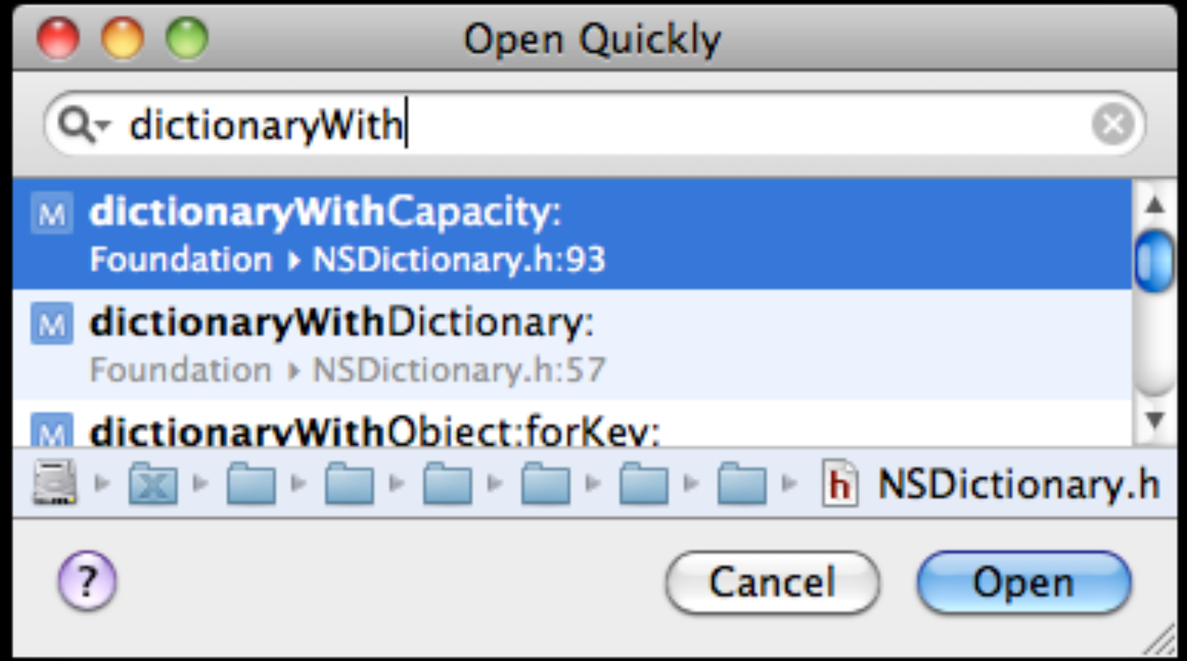
- If you have custom bindings, you don’t get new bindings with Xcode upgrades

Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file

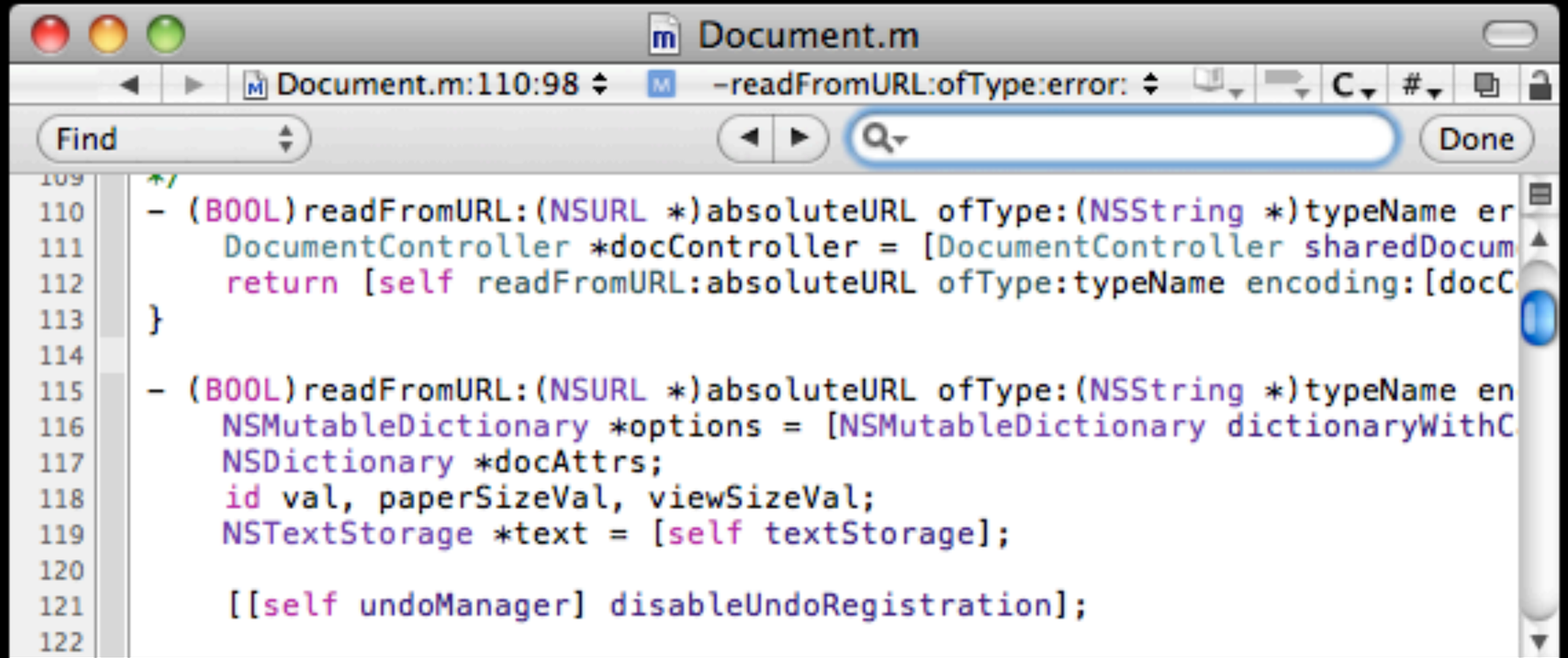
Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file



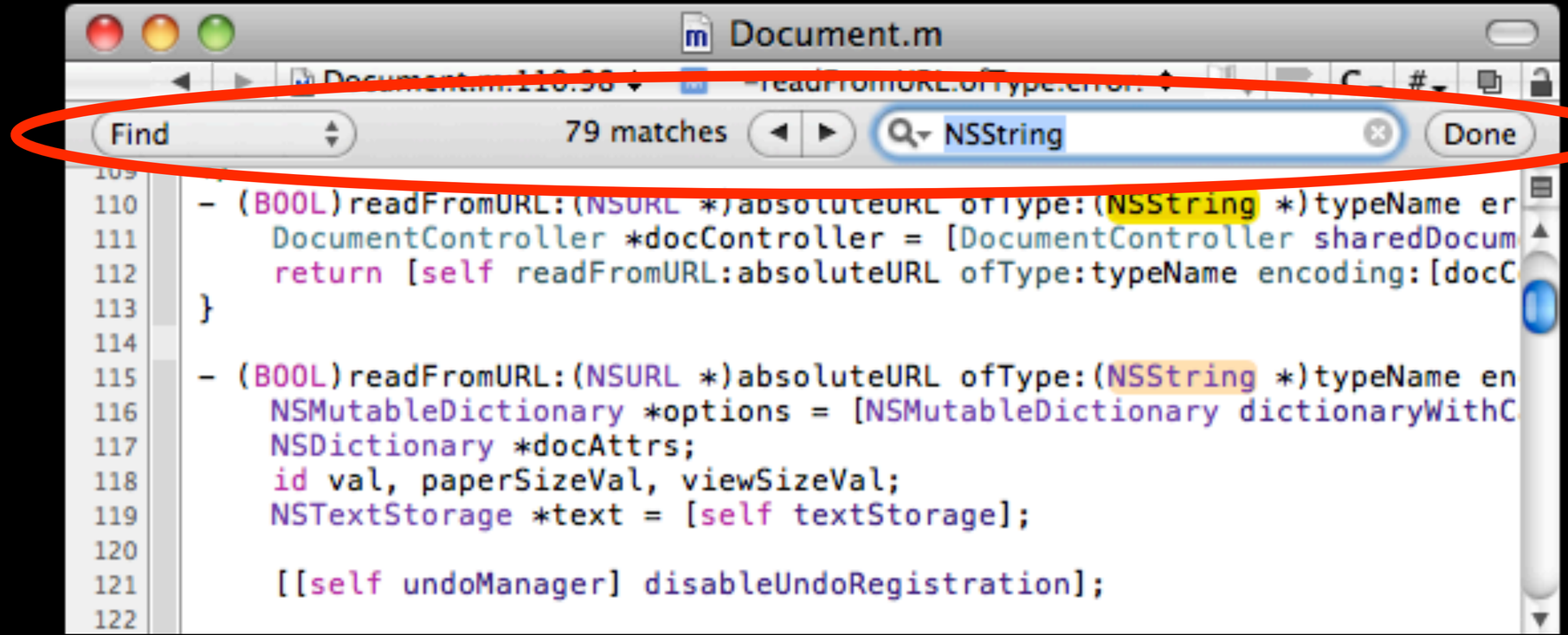
Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file



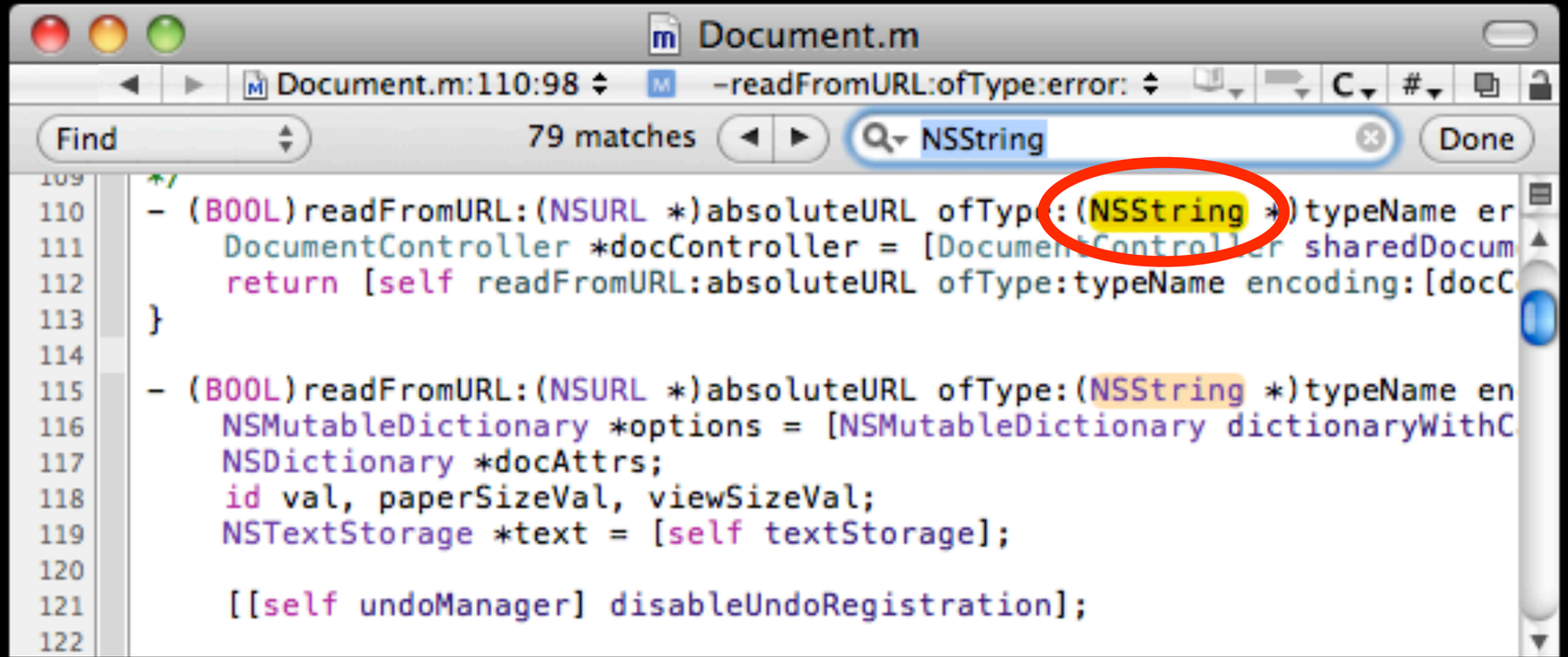
Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file



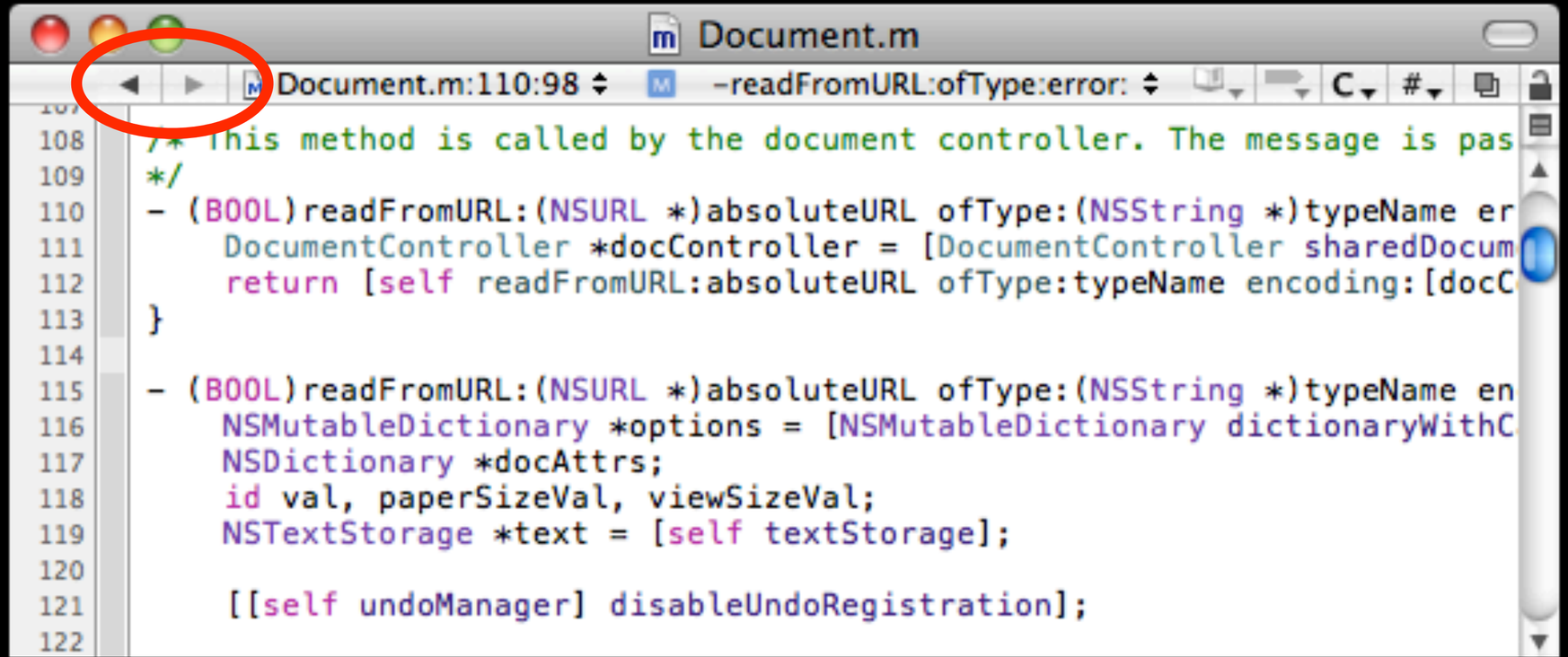
Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file



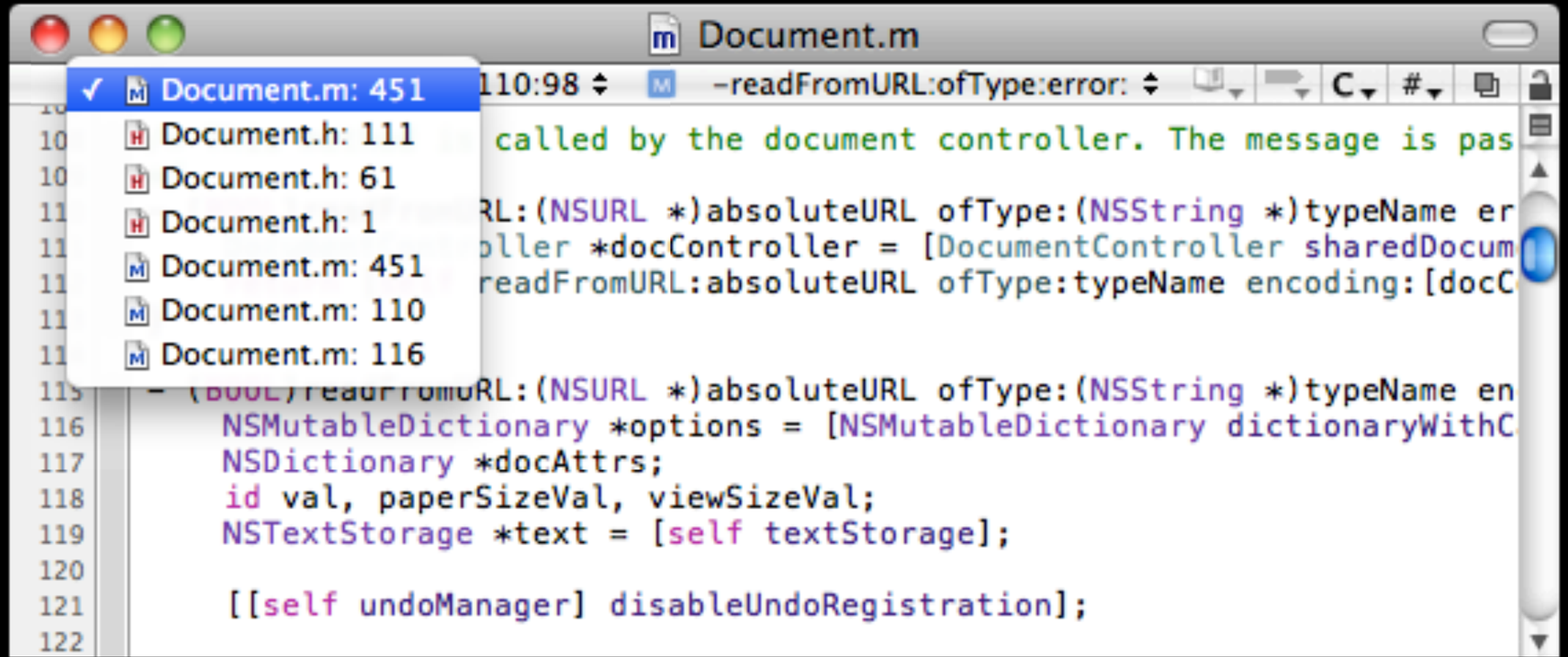
Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file



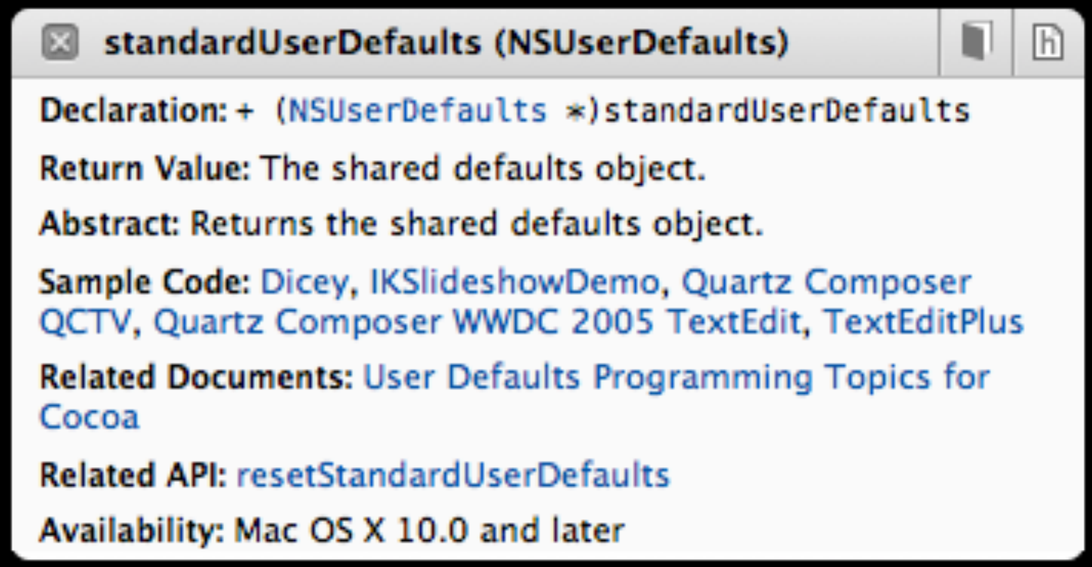
Navigation

- Open Quickly
- Incremental find banner
- History now includes navigation within a file



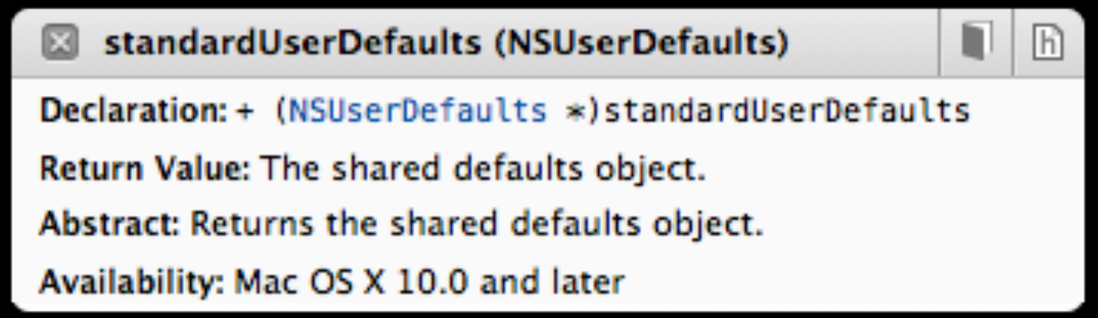
Quick Help

- Small floating API summary
- Can drag it off to make it stick
- Customize what's shown



Quick Help

- Small floating API summary
- Can drag it off to make it stick
- Customize what's shown



Demo

Editing efficiently

Navigation Shortcuts

• Go to Definition	`Command-double-click`
• Quick Help	`Option-double-click`
• Go to Documentation	`Command-option-double-click`
• File pop-up	`Control-1`
• Function pop-up	`Control-2`
• Go Back	`Command-option-left arrow`
• Previous File	`Shift-command-option-left arrow`

Incremental Find Shortcuts

• Next match	`Return`
• Previous match	`Shift-Return`
• Hide the find banner and select the text	`Tab`
• Hide the find banner and go back	`Escape`

Customizing the Editor

- Text macros
- User scripts

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
     $(ORGANIZATIONNAME)\n// All rights reserved.\n
     //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
     $(ORGANIZATIONNAME)\n// All rights reserved.\n
     //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
     $(ORGANIZATIONNAME)\n// All rights reserved.\n
     //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
     $(ORGANIZATIONNAME)\n// All rights reserved.\n
     //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
     $(ORGANIZATIONNAME)\n// All rights reserved.\n
     //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
    $(ORGANIZATIONNAME)\n// All rights reserved.\n
    //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

- Defined as property lists

```
{ Identifier = c.copyright;
  IsMenuItem = YES;
  Name = "Copyright Comment";
  TextString = "//\n// Copyright (c) $(YEAR),
     $(ORGANIZATIONNAME)\n// All rights reserved.\n
     //\n";
  CompletionPrefix = "copyright";
};
```

Text Macros

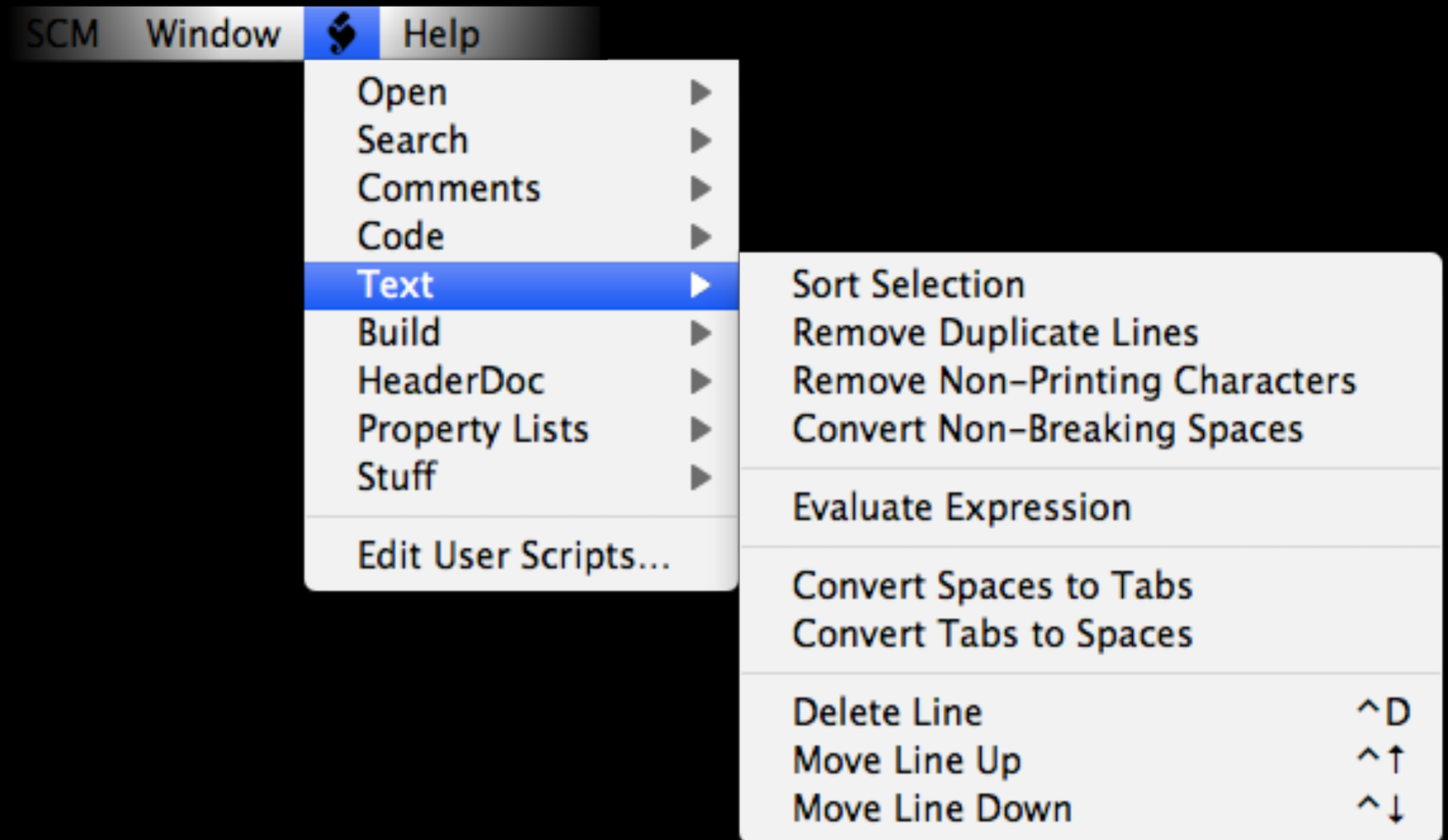
- Files with `.xctxtmacro` extension
- Built-in macros:

```
/Developer/Applications/Xcode.app/Contents/PlugIns/
TextMacros.xctxtmacro
```

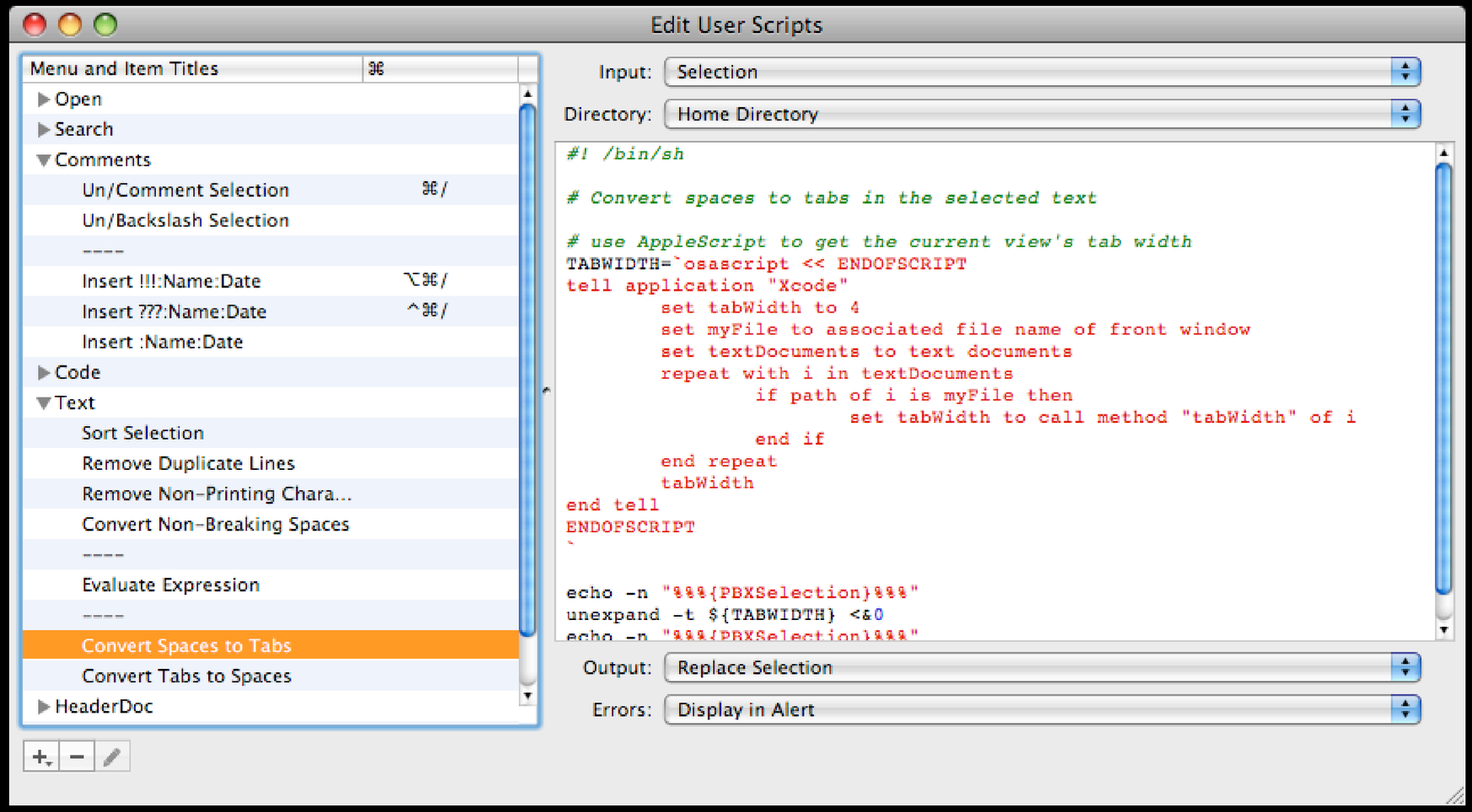
- Install yours into:

```
~/Library/Application Support/Developer/Shared/Xcode/
Specifications
```

User Scripts



Editing User Scripts



Demo

Customizing the editor

Agenda

- Text editing
- Building and debugging
- Source control management and snapshots

Building and Debugging

Chris Hanson
Xcode Engineer

Building

- Build settings, phases, and rules (oh my!)
- Parallel and distributed builds
- Automating builds

Building

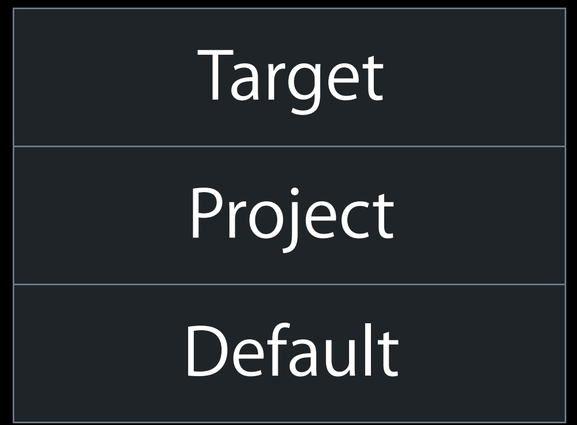
How does Xcode build your software?

- Project: Container—it doesn't build anything
  - Like a "solution" or "workspace" in other IDEs
- Target: Builds a product like an app, plug-in, or framework
  - Build phases structure the build
  - Build rules perform the build
  - Build settings are knobs and switches

Building

Build settings

- Parameters to your build
- Key-value pairs
- Collected in configurations per layer



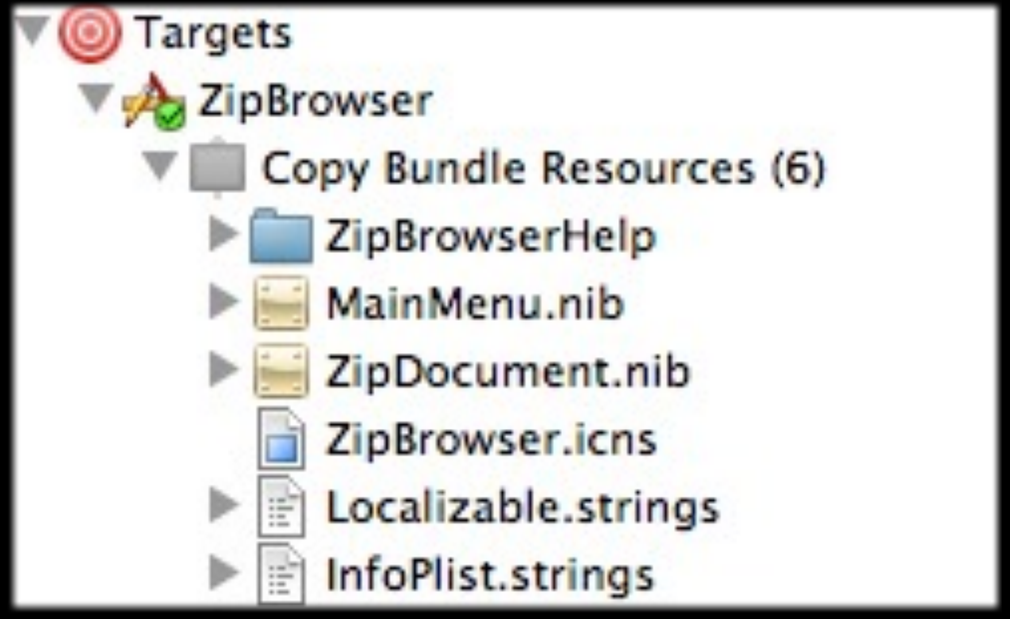
Documentation

Xcode Build Settings Reference

Building

Copy Files build phases

- Get “Copy Bundle Resources” by default
- Copy to other paths within your built product
- Embed plug-ins and frameworks



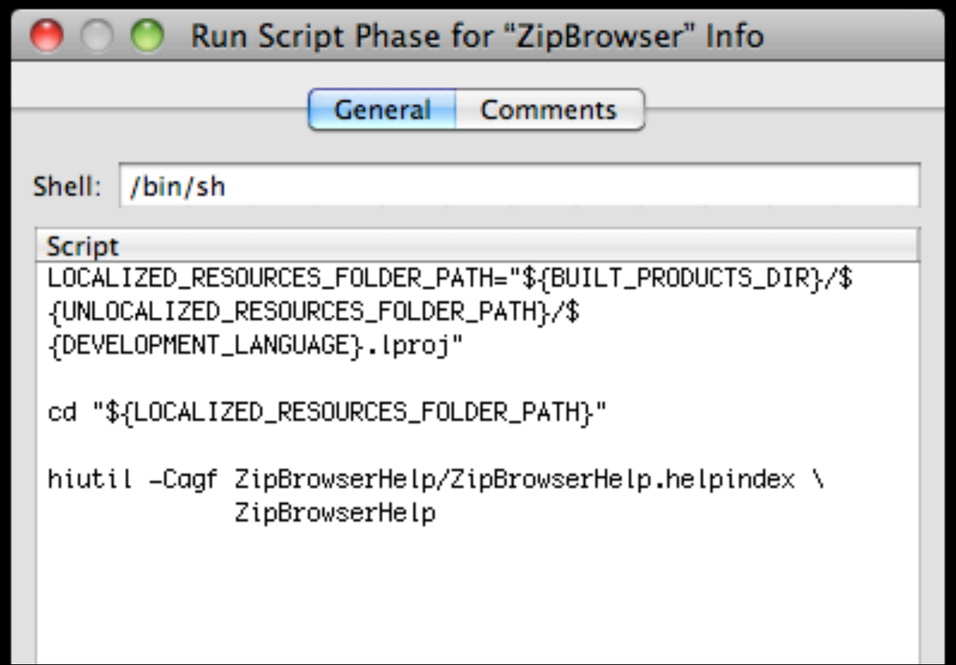
Building

Run Script build phases

- Run arbitrary scripts or tools in your build
- Xcode parses error and warning information
- Participate in dependency management

Pro Tip

Keep scripts in separate files in your project.



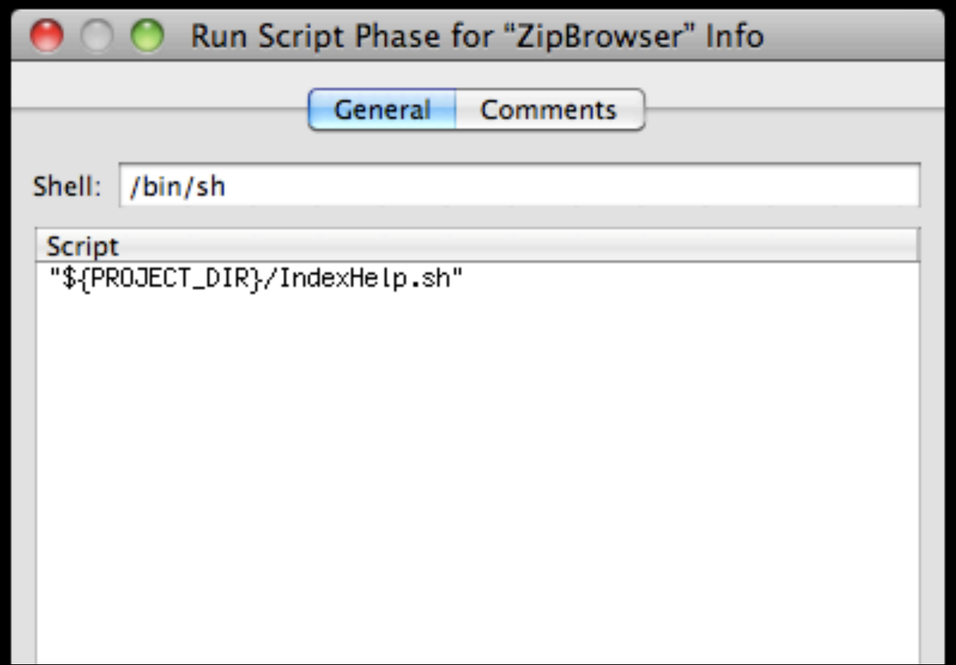
Building

Run Script build phases

- Run arbitrary scripts or tools in your build
- Xcode parses error and warning information
- Participate in dependency management

Pro Tip

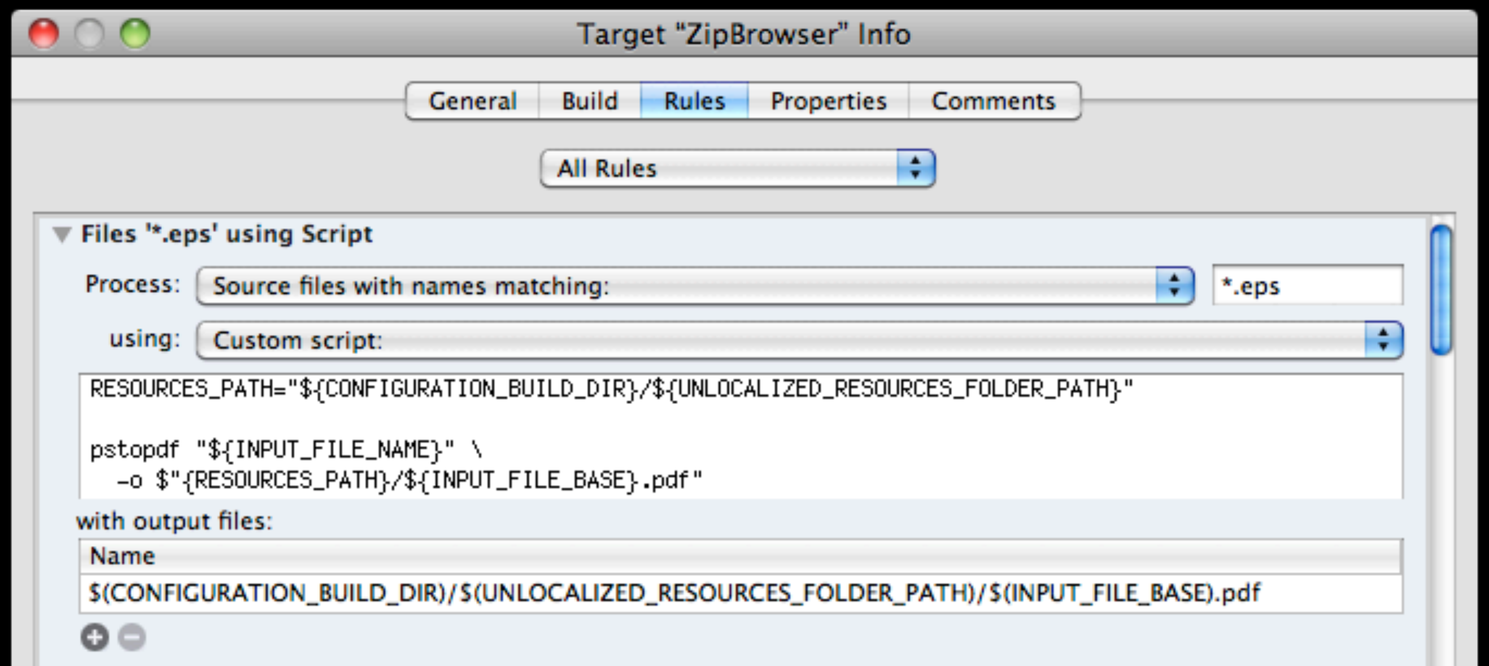
Keep scripts in separate files in your project.



Building

Custom build rules

- Transform files matching a pattern into an output using a script
- Xcode parses error and warning information



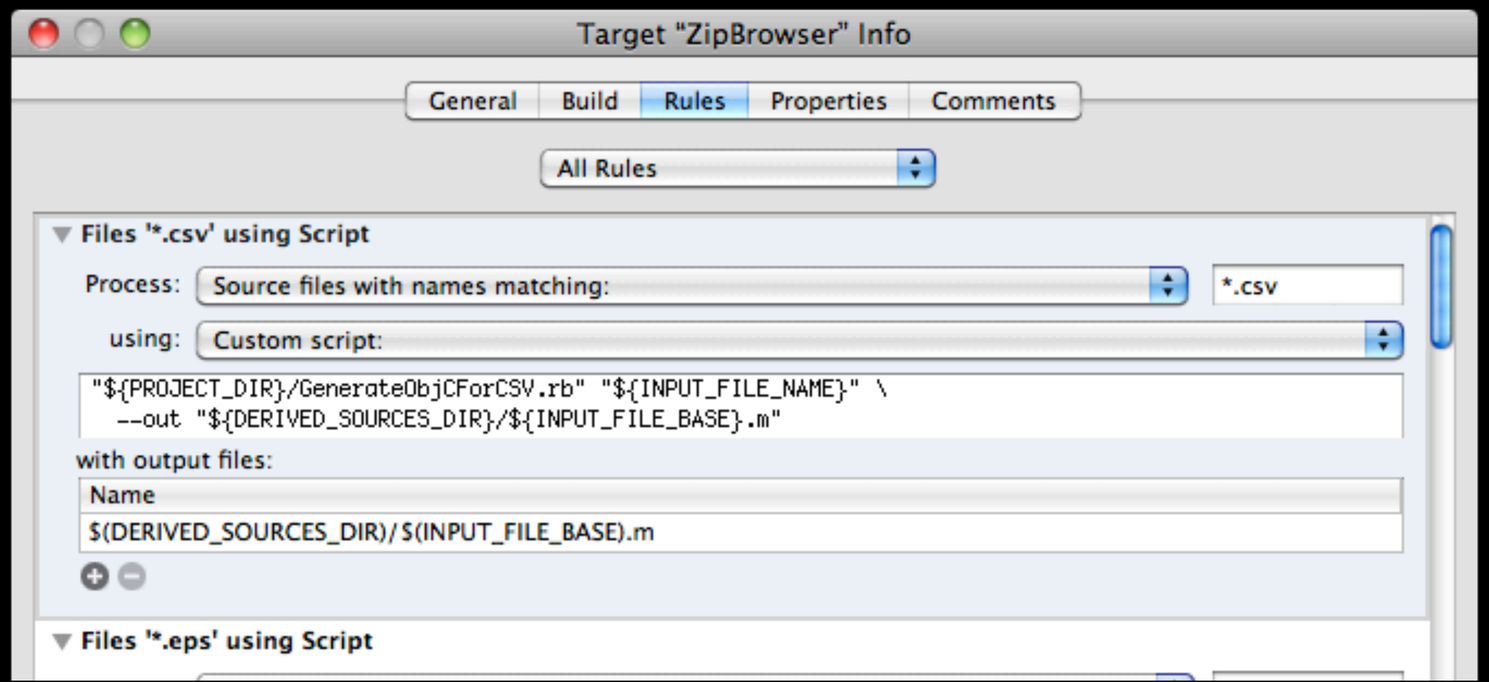
Pro Tip

Keep scripts in separate files in your project.

Building

Custom build rules

- Output can be further processed by other rules
  - Compiled and linked automatically
  - *Any* matching rule will be followed



Pro Tip

Keep scripts in separate files in your project.

Building

Parallel and Distributed Builds

- Xcode leverages multiple cores whenever it can
- Build independent targets in parallel
- Distributed Builds use Macs on your network

Building

Automating builds

- Use the `xcodebuild` tool to build an Xcode project
- Specify target, configuration, SDK, build settings
- Distribute builds via Bonjour or to specific hosts

```
$ xcodebuild ZipBrowser.xcodeproj \
    -target ZipBrowser \
    -configuration Release \
    ARCHS="i386 x86_64"
```

Demo

Building

Debugging

- Configuring the debugger interface
- Data formatters
- Executables
- The command-line debugger, `gdb`

Debugging

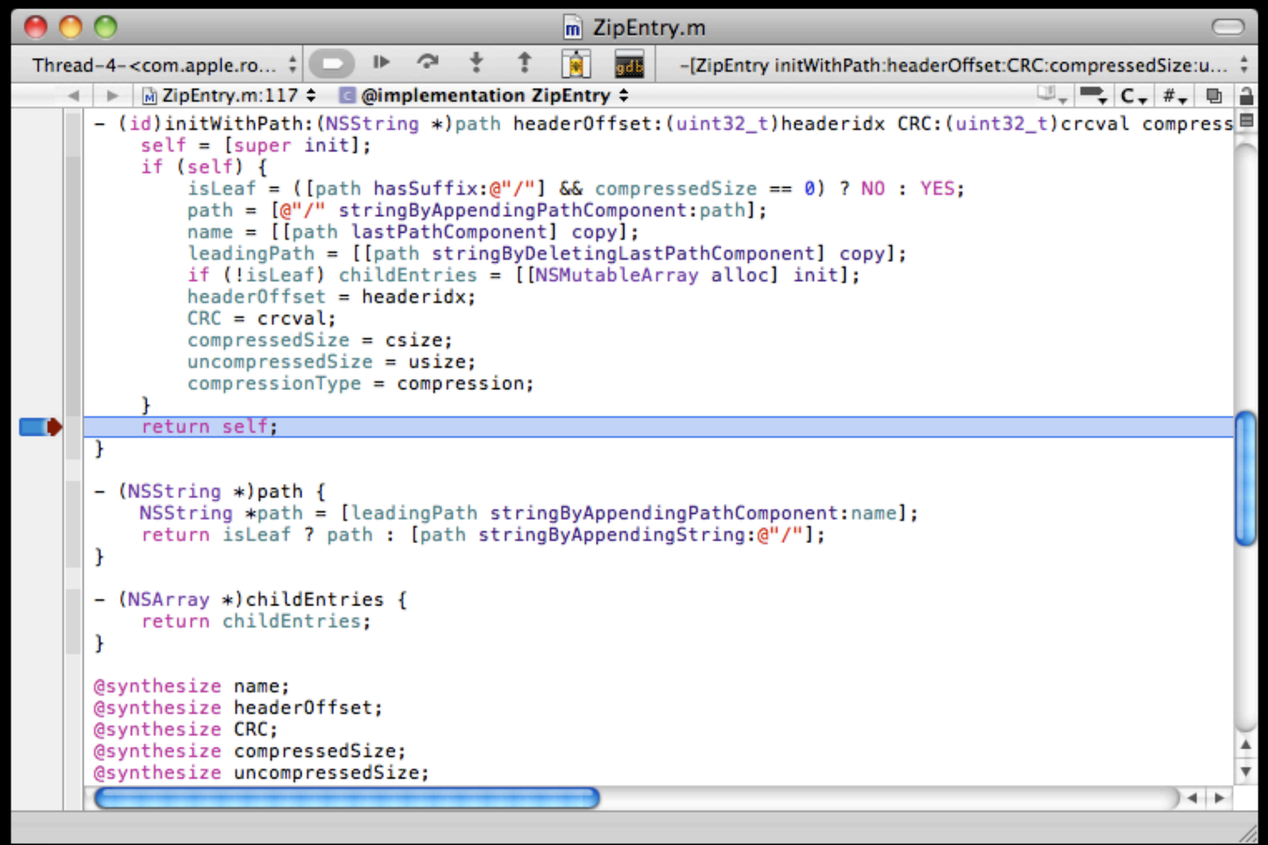
Configuring the debugger interface

- In-editor debugger controls
- Vertical interface layout
- Source and assembly views

Debugging

Configuring the debugger interface

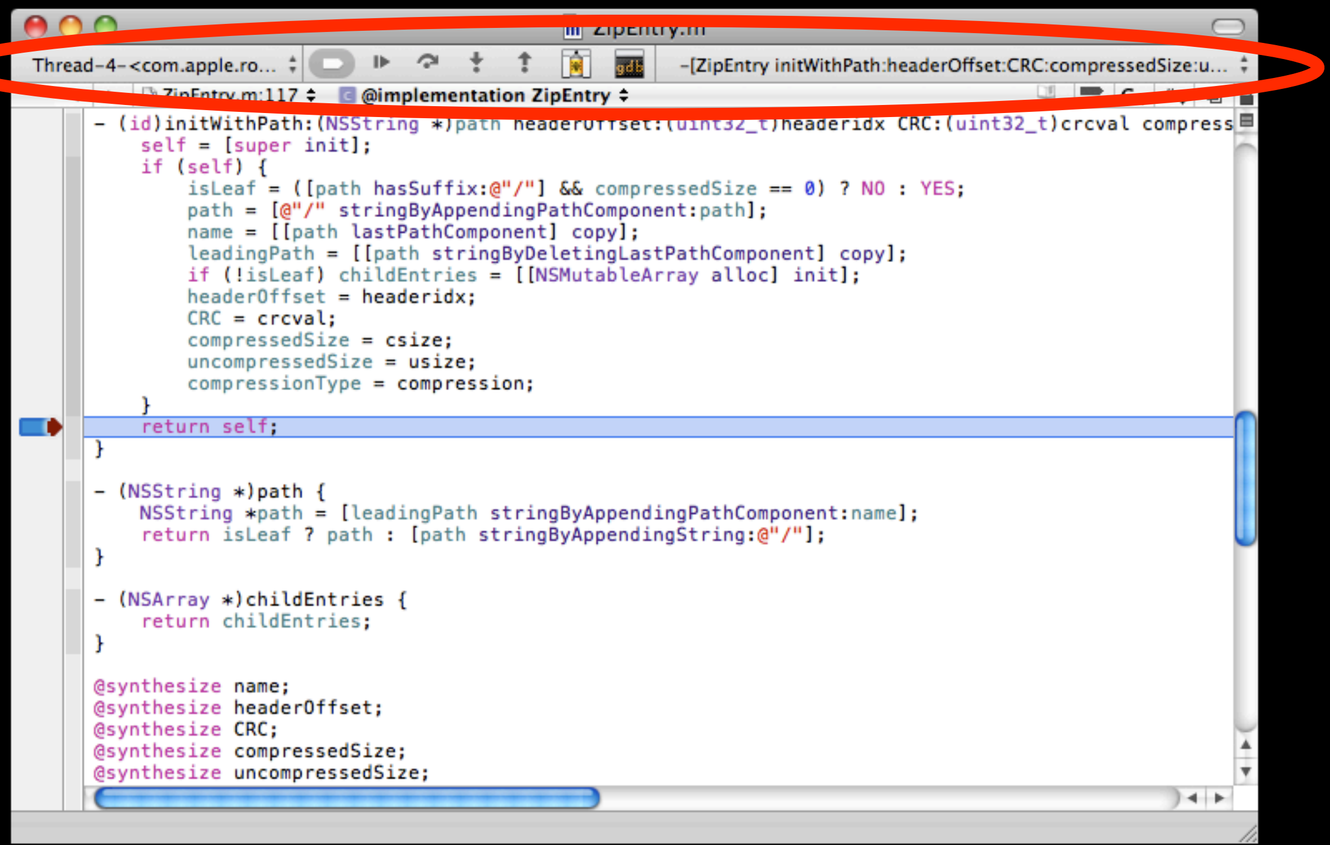
- In-editor debugger controls
- Vertical interface layout
- Source and assembly views



Debugging

Configuring the debugger interface

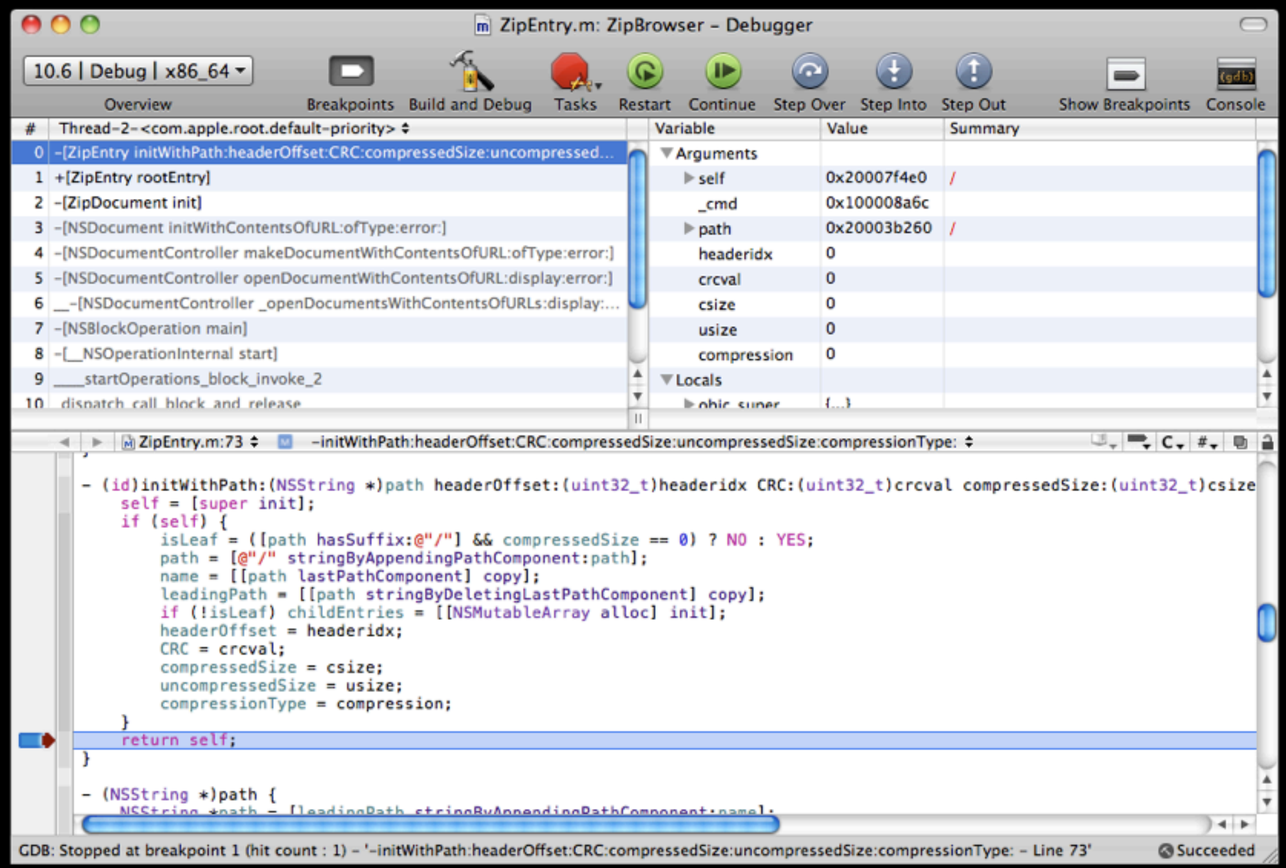
- In-editor debugger controls
- Vertical interface layout
- Source and assembly views



Debugging

Configuring the debugger interface

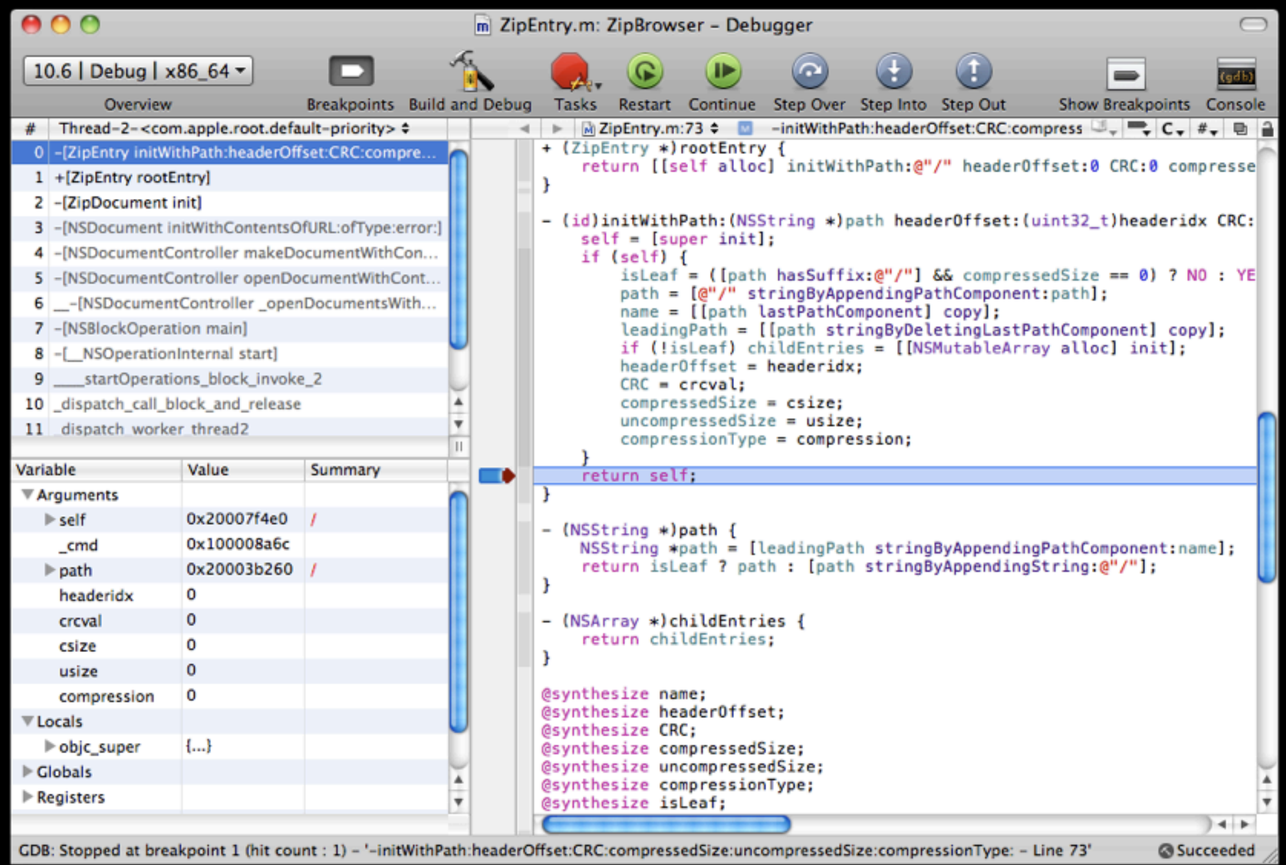
- In-editor debugger controls
- Vertical interface layout
- Source and assembly views



Debugging

Configuring the debugger interface

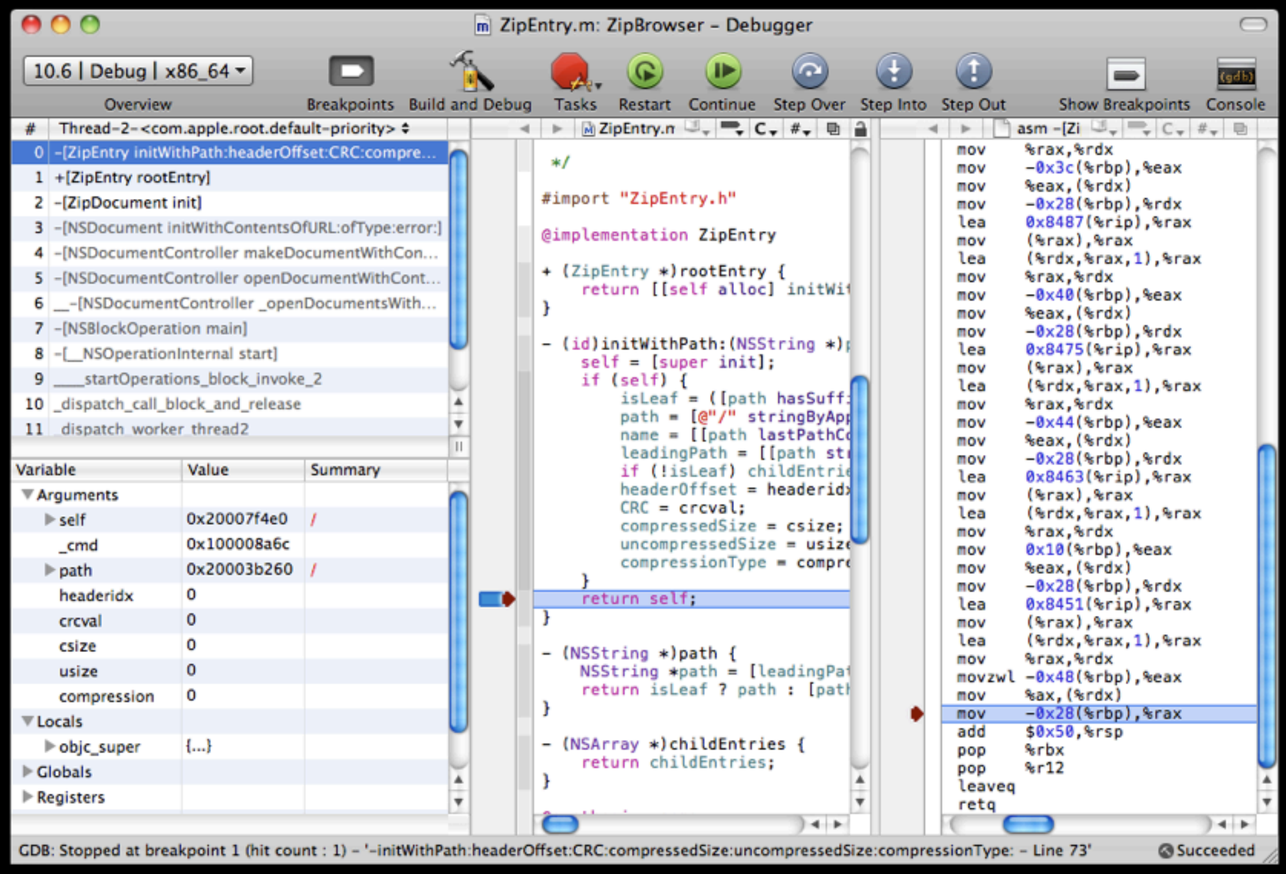
- In-editor debugger controls
- Vertical interface layout
- Source and assembly views



Debugging

Configuring the debugger interface

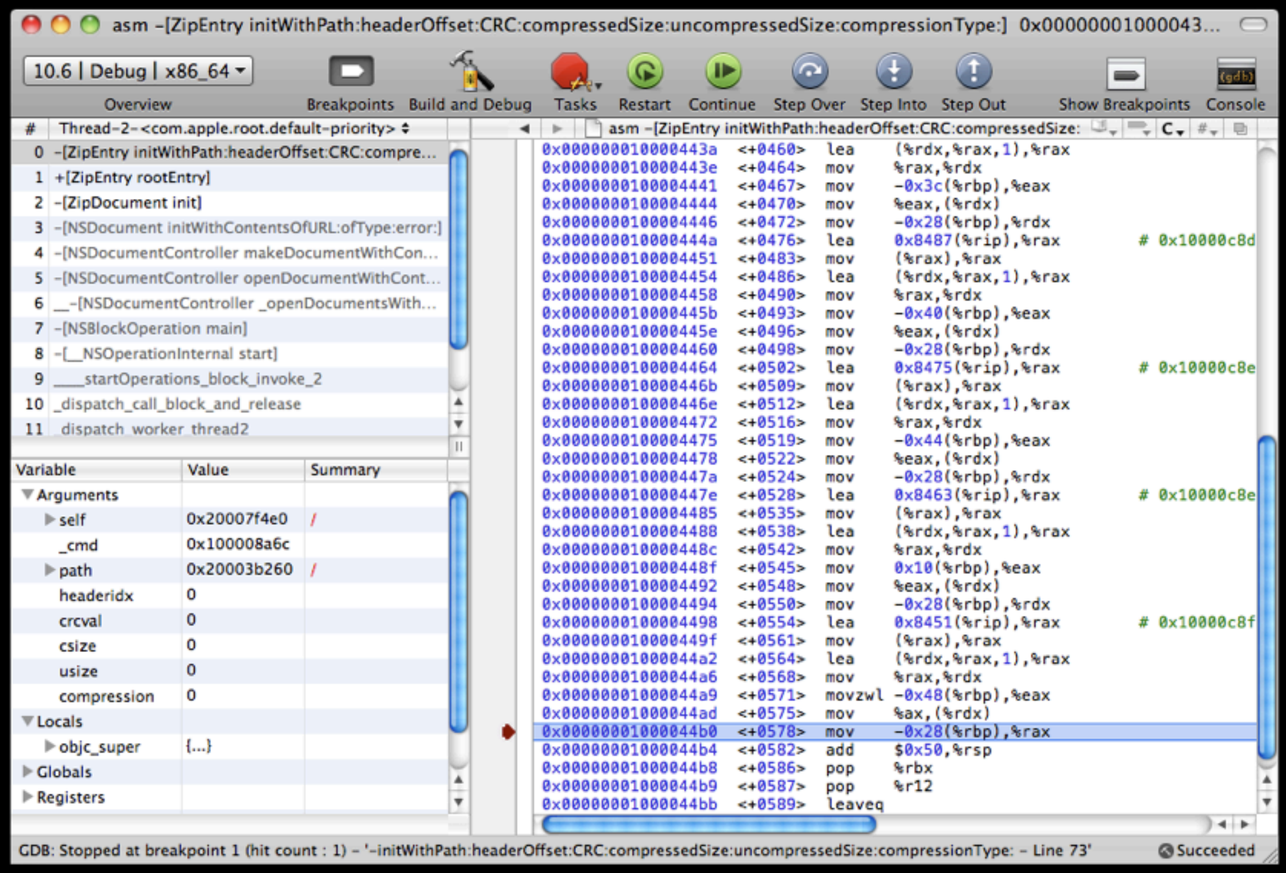
- In-editor debugger controls
- Vertical interface layout
- Source and assembly views



Debugging

Configuring the debugger interface

- In-editor debugger controls
- Vertical interface layout
- Source and assembly views



Debugging

Data formatters

- Show custom information for your own types
- Debugger expressions derive what's shown

```
{(NSString *)[$VAR valueForKey:@\"name\"]}:s
```

Documentation

Xcode Debugging Guide

Debugging

Data formatters

- Show custom information for your own types
- Debugger expressions derive what's shown

Variable	Value	Summary
▼ Arguments		
▶ self	0x20005eb60	{(NSString *)[$VAR name]}:s
_cmd	0x100008a6c	
▶ path	0x200084860	/ZipBrowser/ZipBrowserAfter/Acce:
headeridx	3186	
crcval	293582291	
csize	1547	
usize	3246	
compression	8	

Debugging

Data formatters

- Show custom information for your own types
- Debugger expressions derive what's shown

Variable	Value	Summary
▼ Arguments		
▶ self	0x20005eb60	AccessibilityElement.h
_cmd	0x100008a6c	
▶ path	0x200084860	/ZipBrowser/ZipBrowserAfter/Acce:
headeridx	3186	
crcval	293582291	
csize	1547	
usize	3246	
compression	8	

Debugging

Data formatters

- Show custom information for your own types
- Debugger expressions derive what's shown
- Data formatter plug-ins for more complex tasks

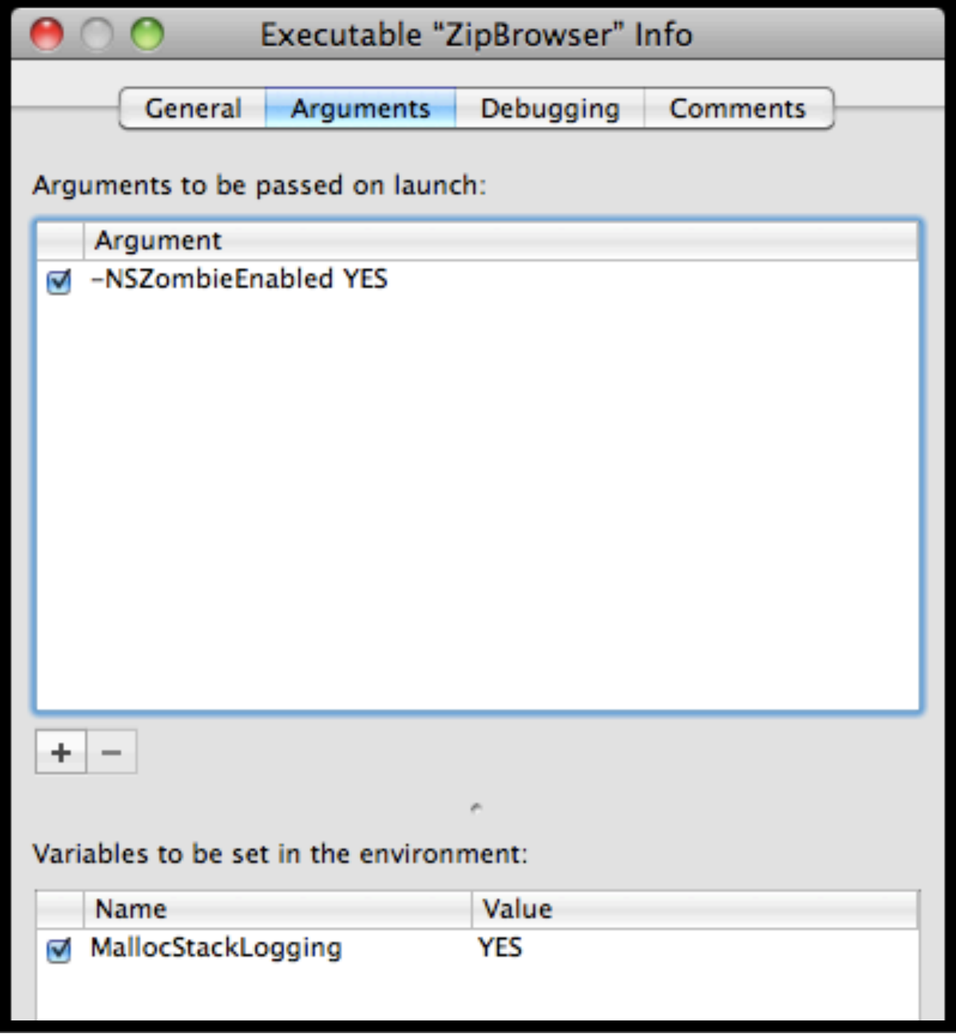
```
Xcode.app/Contents/PlugIns/
    GDBMIDebugging.xcplugin/Contents/
        Headers/DataFormatterPlugin.h
```

- Run by gdb *in your app,* so be careful!

Debugging

Executables

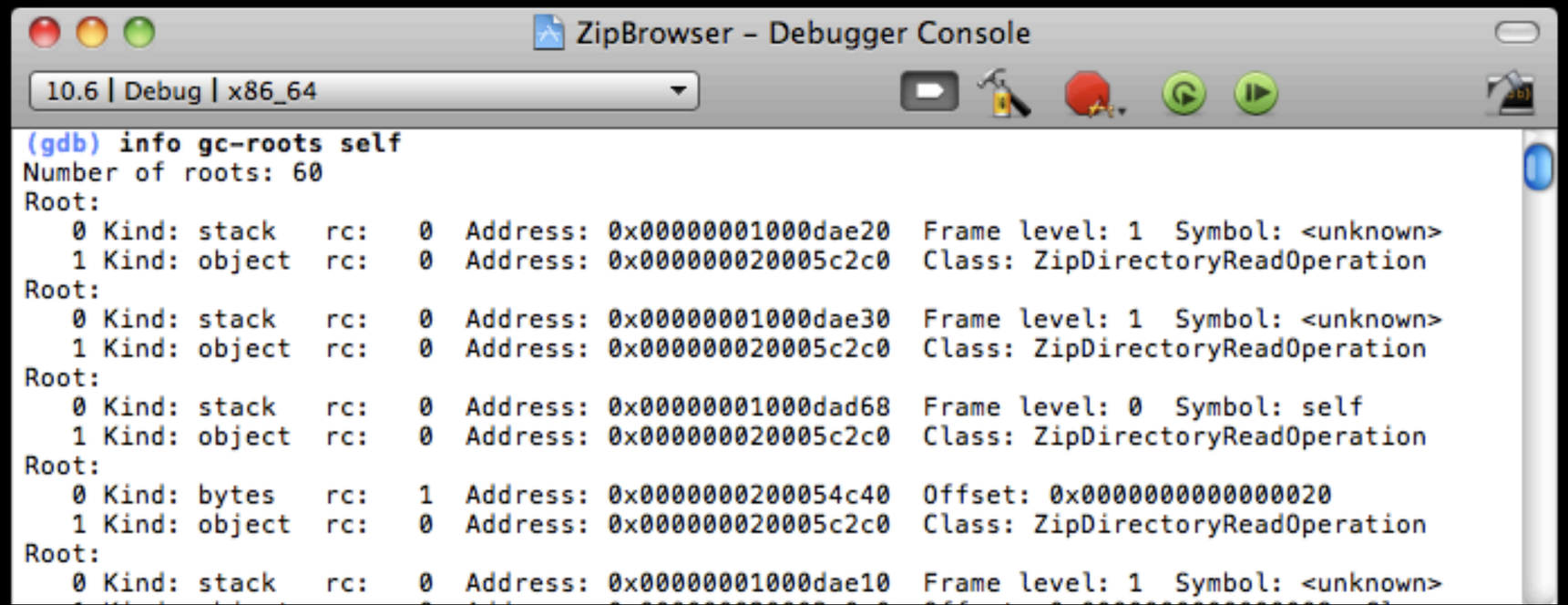
- Tweak how Xcode runs and debugs your app
  - Command-line arguments
  - Environment variables
  - Working directory
- Debug unit tests
- Remote debugging



Debugging

gdb

- Full access to the command language
- Create macros for complex queries



Demo

Debugging

Agenda

- Text editing
- Building and debugging
- Source control management and snapshots

Source Control Management and Snapshots

Andrew Pontious

Source Control Management and Snapshots

- SCM autoconfiguration
- Multiple project roots
- Comparisons with snapshots and SCM

Source Control Management and Snapshots

- SCM autoconfiguration
- Multiple project roots
- Comparisons with snapshots and SCM

Demo

SCM autoconfiguration

Justin Gareau
Xcode QA

Source Control Management and Snapshots

- SCM autoconfiguration
- Multiple project roots
- Comparisons with snapshots and SCM

Source Control Management and Snapshots

- SCM autoconfiguration
- Multiple project roots—new in Snow Leopard
- Comparisons with snapshots and SCM

Source Control Management and Snapshots

What is a project root?

Slides 73 - 85 treated as a sequence build. Master used is graphic format because dominant slides are graphic.

- MSH

Source Control Management and Snapshots

What is a project root?

Source Control Management and Snapshots

What is a project root?

Source Control Management and Snapshots

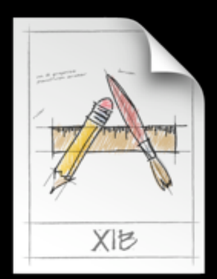
What is a project root?

Source Control Management and Snapshots

What is a project root?

Source Control Management and Snapshots

What is a project root?



Source Control Management and Snapshots

What is a project root?

- Source control
- Snapshots
- Symbolic Project Find

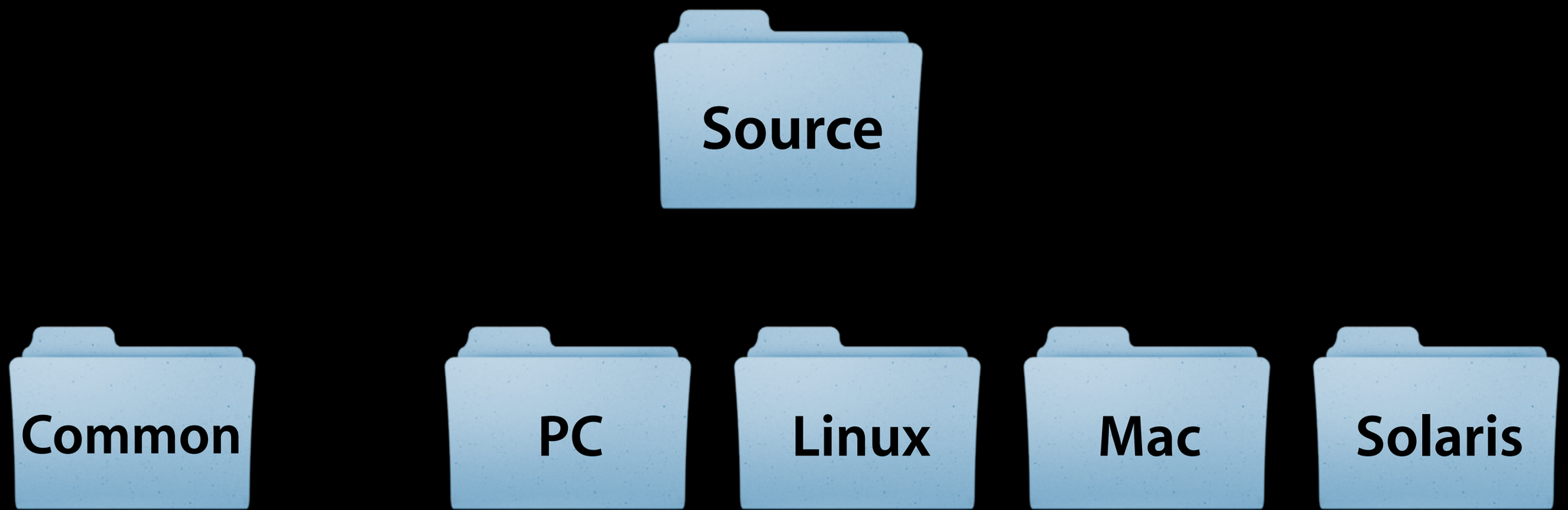
Source Control Management and Snapshots

What is a project root?

- A directory containing all files in project
- Used by directory-based features
  - SCM
  - Snapshots
  - Symbolic Project Find
- Simple and easy to use

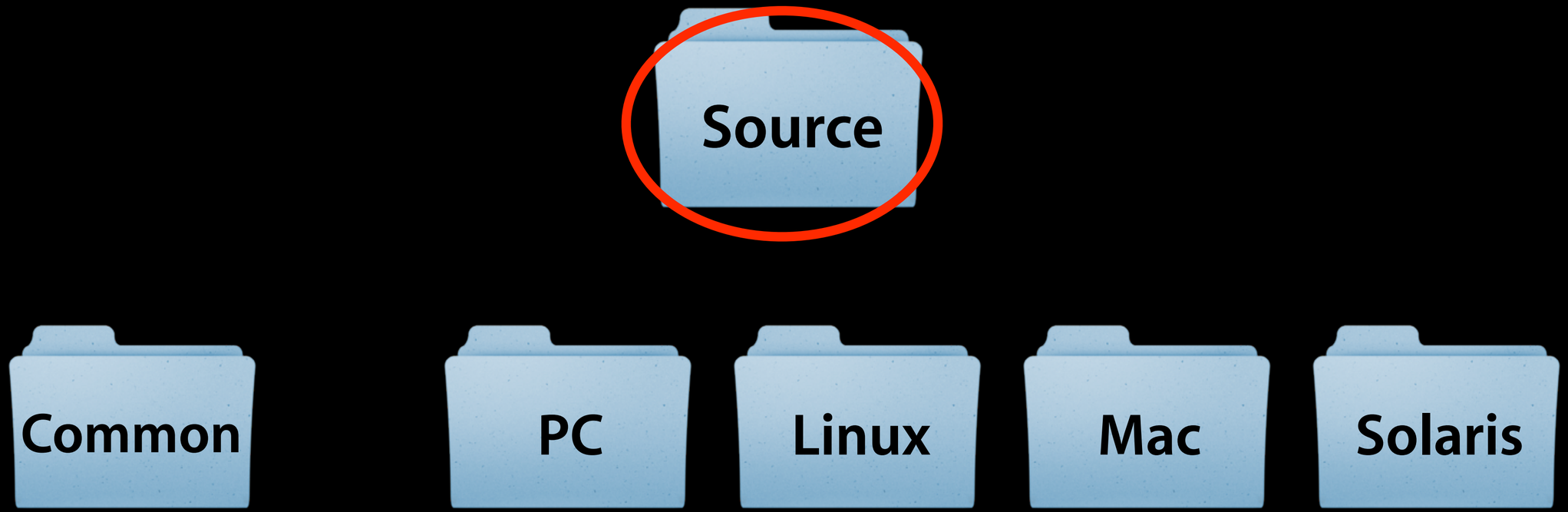
Source Control Management and Snapshots

Multiple project roots



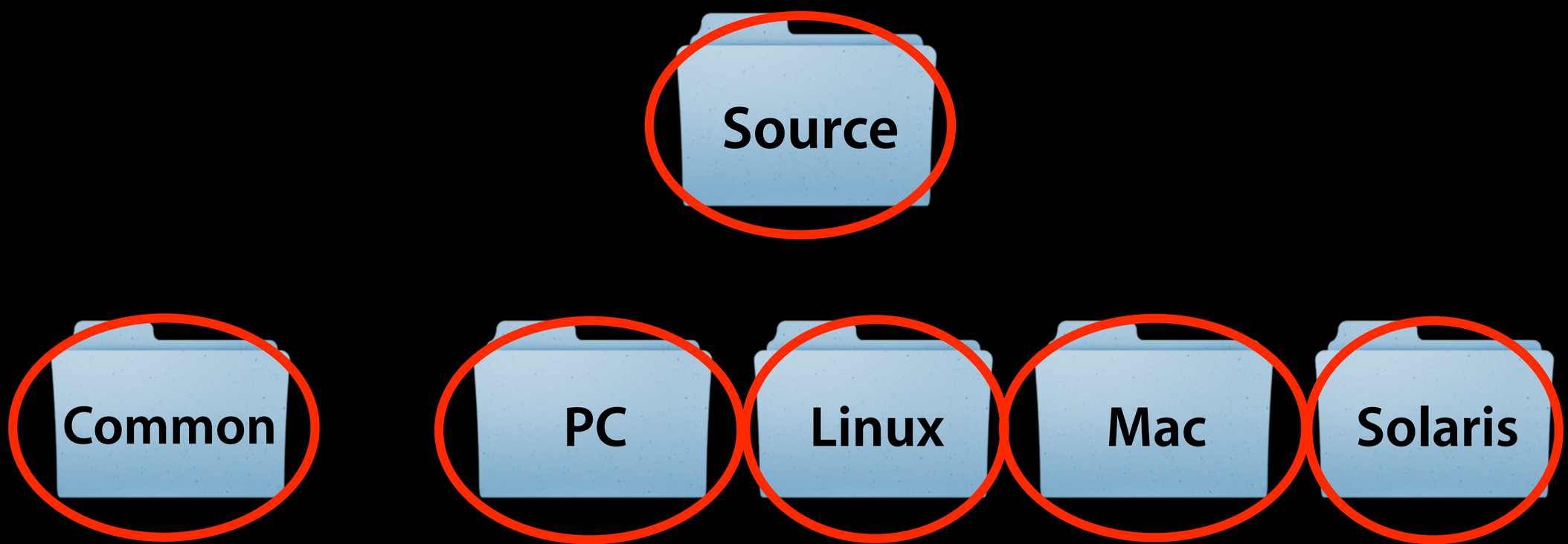
Source Control Management and Snapshots

Multiple project roots



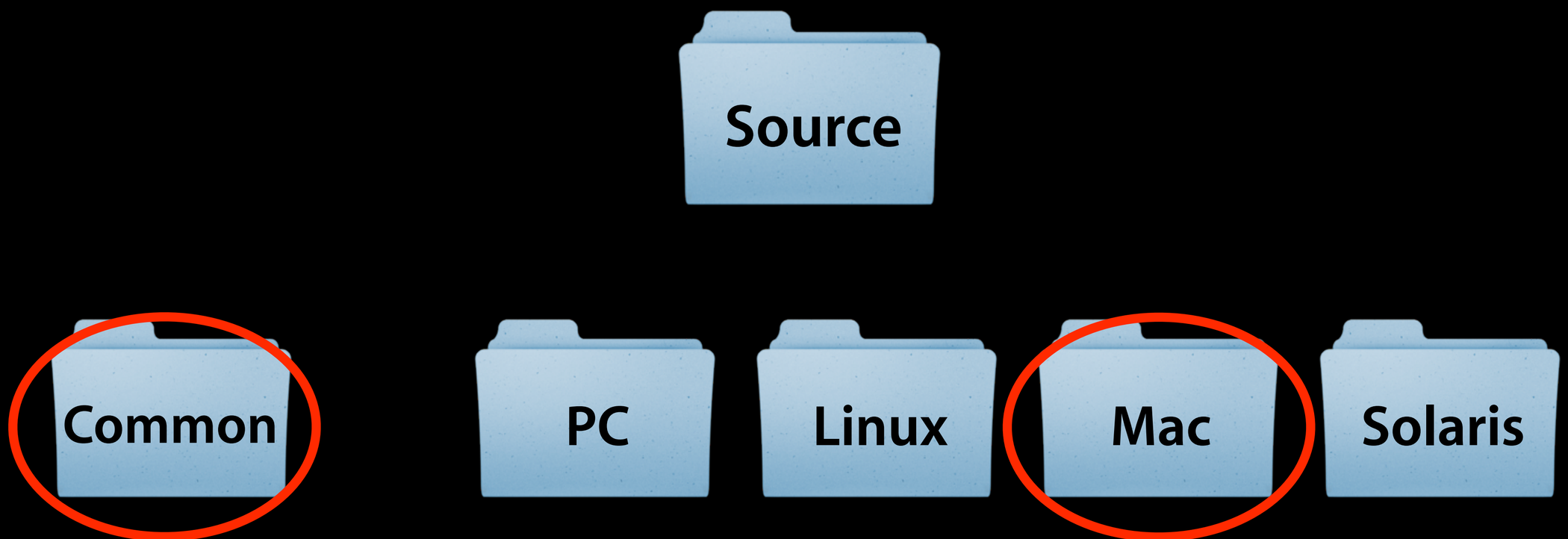
Source Control Management and Snapshots

Multiple project roots



Source Control Management and Snapshots

Multiple project roots



Source Control Management and Snapshots

Multiple project roots

Demo

Multiple project roots

Source Control Management and Snapshots

- SCM autoconfiguration
- Multiple project roots
- Comparisons with Snapshots and SCM

Source Control Management and Snapshots

- SCM autoconfiguration
- Multiple project roots
- Comparisons with Snapshots and SCM
  - Snapshots and nib comparison
  - SCM and project file comparison

Source Control Management and Snapshots

What are snapshots?

Source Control Management and Snapshots

What are snapshots?

⌘Z

Source Control Management and Snapshots

What are snapshots?

⌘Z

Source Control Management and Snapshots

What are snapshots?

⌘Z

Snapshots

Source Control Management and Snapshots

Snapshot location

- ~/Library/Application Support/Developer/SharedSnapshotRepository.sparseimage
- XCSnapshotDiskImagePath

Source Control Management and Snapshots

Snapshot location

- ~/Library/Application Support/Developer/SharedSnapshotRepository.sparseimage
- XCSnapshotDiskImagePath

Documentation

Xcode Source Management Guide
Xcode User Default Reference

Demo

Comparisons with snapshots and SCM

Summary

- Tune to your liking
  - Text editing
  - Build phases
- Show where and how
  - Snapshot location
  - xcodebuild
- More efficient
  - Multiple roots
  - Debugger options

More Information

Michael Jurewitz
Developer Tools Evangelist
jurewitz@apple.com

Related Sessions

Advanced Debugging and Performance Analysis	Nob Hill Thursday 9:00AM
Managing the iPhone Development Process	Russian Hill Thursday 10:30AM

Labs

Xcode Tools	iPhone Lab B Thursday 10:30AM
Xcode Tools	OS X Lab A Friday 9:00AM

Q&A

WWDC Session Survey

You can rate your sessions on the WWDC Attendee Site.

Survey forms are located on each session description page at:

developer.apple.com/wwdc/attendee

WWDC09